Remarks

The above Amendments and these Remarks are in reply to the Office Action mailed August 5, 2008. Claims 1-13, 15-20, 32-34 were pending in the Application prior to the outstanding Office Action. Claims 1-3 and 18-20 are being amended. Claims 15-17 and 32-24 are being canceled (claims 4-14, 21-31, 35-38 and 39-42 were previously canceled). Thus, claims 1-3 and 18-20 remain for the Examiner's consideration. In view of the above amendments and the following remarks, reconsideration and withdrawal of the outstanding rejections are respectfully requested.

I. Rejections under 35 U.S.C. 112, second paragraph

Claims 1-13, 15-20 and 32-34 were rejected under 35 U.S.C. 112, second paragraph, as allegedly being indefinite for failing to particularly point out and distinctly claim the subject matter which Applicant regards as the invention.

Claims 1 and 18 have been amended in a manner that is believed to overcome this rejection, accordingly it is respectfully requested that this rejection be reconsidered and withdrawn.

II. Rejections under 35 U.S.C. 103(a)

Claims 1-3 and 18-20 were rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over Christensen et al., "Extending Java for High-Level Web Service Construction" (hereinafter Christensen) in view of Meredith (U.S. Patent No. 6,516,322; hereinafter Meredith), and further in view of Peltz, "Web Services Orchestration" (hereafter Peltz).

Claims 15-17 and 32-34 were rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over Christensen in view of Meredith, further in view of Peltz, and further in view of van der Aalst et al. "XML Based Schema Definition for Inter-Organization Workflow", (hereafter Aalst).

II. Discussion of Claims

As is explained in the specification, an embodiment of Applicant's invention (including the embodiment of claims 1 and 18) enables developers that are used to using a specific programming language (e.g., JAVA) because they are familiar with that programming language, to extend the programming language by adding workflow constructs (e.g., an action construct) that are missing from the programming language (e.g., JAVA) but desirable. The method of claim 1 and the system of claim 18, for example, can enable developers that are used to working with Java to add workflow definitions using a workflow language in the form of annotations to the source code and the classes of JAVA source code. Such "annotations" would normally be ignored when the source code was processed, because "annotations" normally are not part of the program itself, and they normally have no direct effect on the operation of the source code they annotate. However, in this invention, such annotations (e.g., created using a workflow language, and provided, e.g., as XML commands) are read by the system and are used to create a workflow program, rather than being ignored. Claims 1 and 18 are limited to adding a specific workflow construct, which is an action construct.

Applicant admits that workflow language and workflow constructs, including action constructs, are not new. Applicant also admits that program source files that include source code with classes therein are not new. Nevertheless, Applicant believes that the invention of claims 1 and 18 provides a novel and unobvious way to use annotations to readily and easily add workflow processing (including action constructs) to source code, such as a Java source file or a Web Service source code, that does not otherwise provide for such workflow processing.

In view of the above, and the discussion below, Applicant asserts that the cited references Christensen, Meredith and Peltz, alone or in combination, do not teach or suggest a "program source file [that] includes a source code and classes therein and a workflow definition created using the workflow language that is specified in the form of annotations to the source code and the classes, and wherein the workflow language extends the source code with a plurality of workflow constructs, including an action construct representing an activity that allows a first software component to call an operation on a second software component", as required by claims 1 and 18 as amended.

In the first full paragraph on page 5 of the Office Action, it is admitted that Christensen and Meredith do not teach a program source file including a source code and classes therein and

a workflow definition that is specified in the form of annotations to the source code and the classes. However, in the second full paragraph on page 5 of the Office Action, it was alleged that Peltz discloses the deficiencies of Christensen and Meredith. More specifically, it was asserted in the Office Action that the BEA WebLogic Workshop section on page 10 of Peltz disclosed "a program source file including a source code and classes written in Java including a workflow created using the existing object oriented programming language that is specified in the form of annotations to the source code and the classes." Applicant respectfully disagrees, as explained below.

On page 10 of Peltz, it is stated that "Conversations within Workshop can be built with JavaDoc annotations tags. For example, to start a conversation, a developer would include the tag, '@jws:conversation: phase=start'" (see page 10 of Peltz, 3rd full paragraph). In summary, this portion of Peltz appears to state that a conversation can be started using a JavaDoc annotation tag. However, a "conversation" is a way to identify a two-way communication, e.g., between one client application and one Web Service so that messages are always returned to the correct client. More specifically, a conversation is a mechanism that causes a response to a request to be returned to the correct requestor.

In contrast, a workflow is a set of steps or actions that occur in a particular order, and in response to specific conditions, in order to perform a task. Thus, there is a clear difference between a "conversation" and a "workflow". Accordingly, Peltz's statement that a JavaDoc annotation tag can be used to "start a conversation" does not teach or suggest that a workflow definition can be created using a workflow language that is specified in the form of annotations to source code and classes of a program source file, as is required by claims 1 and 18.

For at least the above reasons, Applicant respectfully asserts that claims 1 and 18 are not obvious in view of Christensen, Meredith and Peltz, alone or in combination. Accordingly, reconsideration and withdrawal of the 103(a) rejection of claims 1 and 18 are respectfully requested. In the event that the Examiner does not withdraw this rejection, Applicant respectfully requests that the Examiner explain in more detail how he is reading Peltz to teach the above mentioned deficiencies of Christensen and Meredith.

Claims 2-3 depend from and add additional features to claim 1. Claims 19-20 depend from and add additional features to claim 18. Applicant respectfully asserts that these claim are

patentable for at least the reason that they depend from independent claims 1 or 18, as well as for the features that they add.

III. Conclusion

In view of the above amendments and remarks, it is respectfully submitted that all of the claims now pending in the subject patent application should be allowable, and reconsideration thereof is respectfully requested. The Examiner is respectfully requested to telephone the undersigned if he can assist in any way in expediting issuance of a patent.

The Commissioner is authorized to charge the required fees and any underpayment of fees or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this reply, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: October 6, 2008

By: /Jeffrey R. Kurin/
Jeffrey R. Kurin
Reg. No. 41,132

FLIESLER MEYER LLP
650 California Street, 14th Floor
San Francisco, California 94108
Telephone: (415) 362-3800
Facsimile: (415) 362-2928
Customer No.: 23910